



09. Manejo de cadenas

Programación - 1º DAM

Luis del Moral Martínez

versión 20.10

Bajo licencia CC BY-NC-SA 4.0



Contenidos del tema

1. Manejo de cadenas

1. Manejo de cadenas

Concepto de cadena

- Una **cadena** es un **array de caracteres** (**char**) terminado por el **carácter nulo** **\0**

Array de caracteres

C	a	m	a		
---	---	---	---	--	--

Cadena de caracteres

C	a	m	a		\0
---	---	---	---	--	----

- El número total de caracteres es la longitud de la cadena + 1 (contando el carácter nulo)

1. Manejo de cadenas

Declaración de una cadena

- Las cadenas de caracteres se declaran como un array de char
- También podemos usar el objeto string ([más información](#))
- Ejemplo:

```
char cadena [100];
```

1. Manejo de cadenas

Inicialización de una cadena

- Las cadenas pueden ser inicializadas durante su declaración
- El **compilador** se encarga de añadir el carácter nulo \0
- Una cadena no puede ser inicializada fuera de la declaración
- **Ejemplo:**

```
char cadena [100] = "Hola mundo";  
cadena = "hola"; // ERROR
```

1. Manejo de cadenas

Lectura de una cadena

- Las cadenas se pueden leer con cin

- **Ejemplo:**

```
char nombre[20];  
cin >> nombre;
```

1. Manejo de cadenas

Funciones miembro cin y cout (1)

- Los objetos **cin** y **cout** (flujos de entrada y salida) incluyen las siguientes funciones:
 - **cin.getline()**: lee una cadena de caracteres completa
 - La longitud de la cadena debe contemplar el **\0** y el **\n** del final de la cadena
 - `cin.getline(cadena, longitud+2)`
 - **cin.get()**: lee del flujo de entrada carácter a carácter
 - `cin.get()` devolverá 1 si ha conseguido leer un carácter
 - Cuando se detecta un retorno de carro se `cin.get()` devuelve 0

1. Manejo de cadenas

Funciones miembro cin y cout (2)

- Los objetos **cin** y **cout** (flujos de entrada y salida) incluyen las siguientes funciones:
 - **cout.put()**: escribe un carácter en el flujo de salida
 - **cin.putback()**: restaura el último carácter leído por **cin.get()** de nuevo al flujo de entrada cin
 - **cin.ignore()**: lee uno o más caracteres del flujo de entrada cin y no los procesa (son ignorados)
 - **cin.peek()**: consiste en una combinación de **cin.get()** y **cin.putback()**

1. Manejo de cadenas

La biblioteca string (string.h)

- Hay que tener en cuenta que los arrays y las cadenas **se pasan por referencia**, no por valor
- La biblioteca string contiene una serie de funciones para manejo de cadenas
- **Biblioteca String:** [Más información](#)
- **Analizaremos cada ejemplo de la biblioteca y los usaremos en nuestros códigos**
- Abre los ficheros de ejemplo **09_01_cadenas1.cpp** y **09_02_cadenas2.cpp**

Créditos de las imágenes y figuras

Cliparts e iconos

- **Obtenidos mediante la herramienta web [IconFinder](#)** (según sus disposiciones):
 - Diapositiva 1
 - Según la plataforma IconFinder, dicho material puede usarse libremente (free comercial use)
 - A fecha de edición de este material, todos los cliparts son free for comercial use (sin restricciones)

Resto de diagramas y gráficas

- Se han desarrollado en PowerPoint y se han incrustado en esta presentación
- Todos estos materiales se han desarrollado por el autor
 - Si se ha empleado algún icono externo, este se rige según lo expresado anteriormente